

UNIVERSIDAD ADVENTISTA DE CENTRO AMÉRICA
ESCUELA DE INGIENERIA EN SISTEMAS

Implementación de google maps sdk para Android

Asignación presentada como requisito parcial de la materia:

IS244 PROGRAMACIÓN III

Profesor: LIC. BRAYAN FUNEZ

Por

Jonathan Córdova Rivera, James Picado Ortíz

Fecha 27/03/2019

Introducción

Los servicios de biblioteca de Google Play permiten a los desarrolladores Android conectarse con mucha facilidad y usar las poderosas funciones de Google, y proporcionar compatibilidad a las versiones anteriores para nuevas características.

Una de las funciones más ampliamente utilizadas de Google Play Services, la función Maps incluye todo lo que necesite para mostrar un Google Map en su aplicación, colocar marcadores aprovechar la superficie superior del mapa, ya sea; con imágenes o con figuras simples, añadir capaz de nivel interior, e incluso, mostrar vistas de calles.

Además de la función estándar Maps, Google también a proporcionado una biblioteca de código abierto llamada Map Utils que proporciona funciones adicionales tales como; mapas de calor y agrupación de marcadores.

Contenido

Introducción	2
¿Qué es Google Maps?	4
<i>Configurar Android Studio</i>	5
Iniciar un proyecto nuevo	5
<i>Descargar Y Configurar Google Play Services</i>	6
<i>Diseñar interfaz</i>	7
<i>Crear API de Google</i>	10
<i>Cargar el Mapa</i>	15
<i>Tipos de mapas</i>	19
Biografía	25

¿Qué es Google Maps?

Antes que nada, que como desarrolladores debemos entender de donde surge cada aspecto que es tomado en cuenta para la creación de un nuevo proyecto, en este caso es la implementación de un mapa, y podemos decir que es un tipo de servidor de aplicaciones que proporciona mapas de internet y que pertenece a Alphabet Inc (la empresa matriz de Google).

El nacimiento de Google Maps:

El 8 de febrero de 2005 fue cuando se presentó oficial Google Maps. Se trataba de la versión inicial en versión web y que estuvo en fase beta durante meses hasta que finalmente se integró formando parte de Google Local.

Es un hecho que Google Maps no dio accesibilidad a todo un público de un momento a otro y no tenía tanta información, en abril de 2005 llegó a Europa la primera versión, y con el paso del tiempo fueron implementando datos de carreteras, imágenes obtenidas por satélites e indicaciones de navegación. En junio de ese mismo año se libera la primera API de maps, y como todos sabemos esto nos facilita insertar los mapas de Google en otros servicios. En lo mencionado anteriormente nos referíamos directamente a la versión web, no existían los smartphones tan desarrollados como ahora los conocemos, pero un gran impulso para maps fue llegada de iPhone, luego de esto era obvio que las demás compañías como Android, Google, se vieron como en la responsabilidad de también integran Maps en el sistema.

Configurar Android Studio

Iniciar un proyecto nuevo

Paso 1: Iniciar y configurar el proyecto

Si no tienes un proyecto abierto, Android Studio te muestra la pantalla de bienvenida.

Para crear un proyecto nuevo, haz clic en Start a New Android Studio project.

Si tienes un proyecto abierto, Android Studio muestra el entorno de desarrollo. Para crear un proyecto nuevo, haz clic en File > New > New Project.

En la siguiente ventana, puedes configurar el nombre de tu app, el nombre de paquete y la ubicación de tu proyecto.

Paso 2: Seleccionar factores de forma y el nivel de API

En la siguiente ventana, puedes seleccionar los factores de forma admitidos por tu app, como teléfonos, tablets, TV, Wear y Google Glass. Los formatos seleccionados se convierten en los módulos de la app dentro del proyecto. Para cada formato, también puedes seleccionar el nivel de API para esa app.

Paso 3: Agregar una actividad

En la siguiente pantalla, puedes seleccionar un tipo de actividad para agregar a tu app. En esta pantalla, se muestra un conjunto diferente de actividades para cada formato que seleccionaste antes.

Paso 4: Configurar tu actividad

En la siguiente pantalla, puedes configurar la actividad que deseas agregar a tu app.

Paso 5: Desarrollar tu app

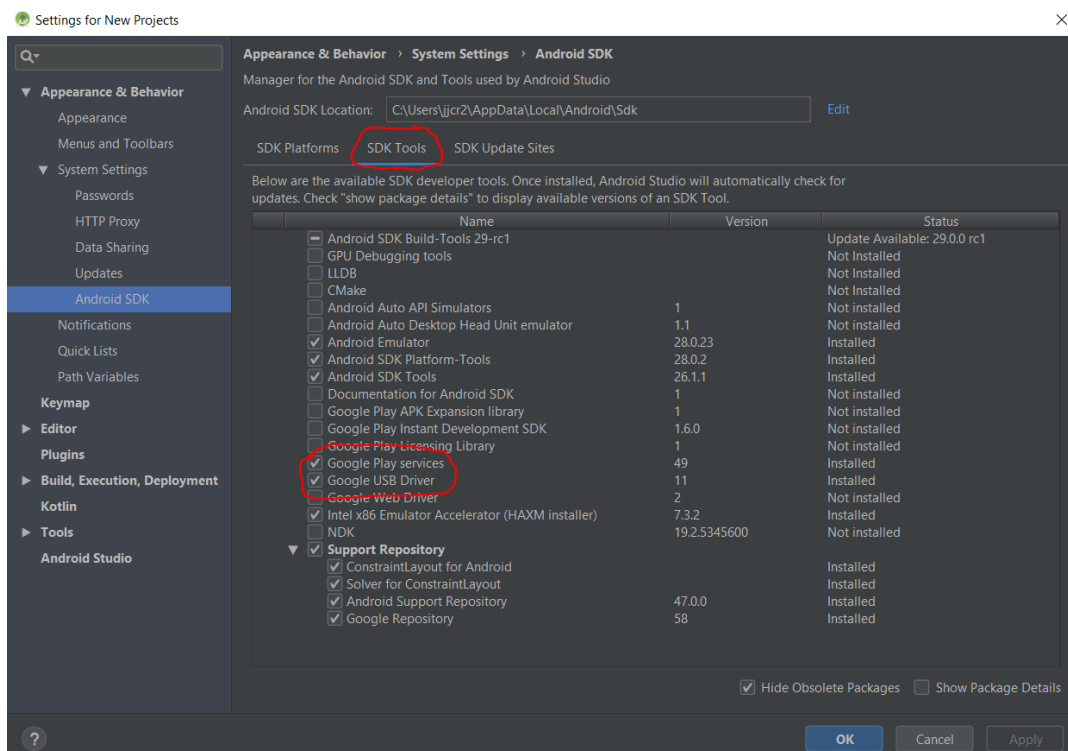
Android Studio crea la estructura predeterminada de tu proyecto y se abre el entorno de desarrollo. Si tu app admite más de un formato, Android Studio crea una carpeta de módulos con archivos de origen completos para cada uno de ellos.

Descargar Y Configurar Google Play Services

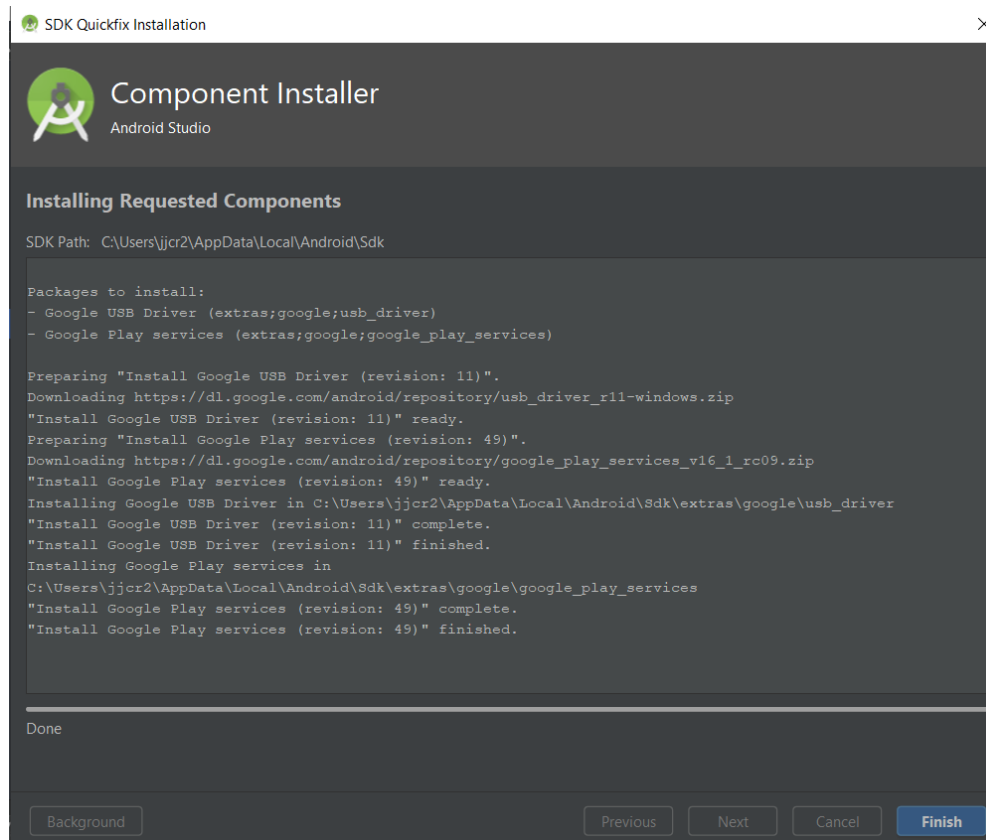
Google Play Services es un conjunto de librerías de Google que brindan a los desarrolladores las funcionalidades de las aplicaciones de Google como Gmail, Analytics, Google Fit, etc. Al igual que Google Maps.

Para agregar la dependencia en nuestro proyecto primero debes instalar el complemento disponible en el SDK.

Dentro de Android Studio dirígete a **Tools > Android > SDK Manager**:



Espera a que termine la descarga y pegado de los archivos:

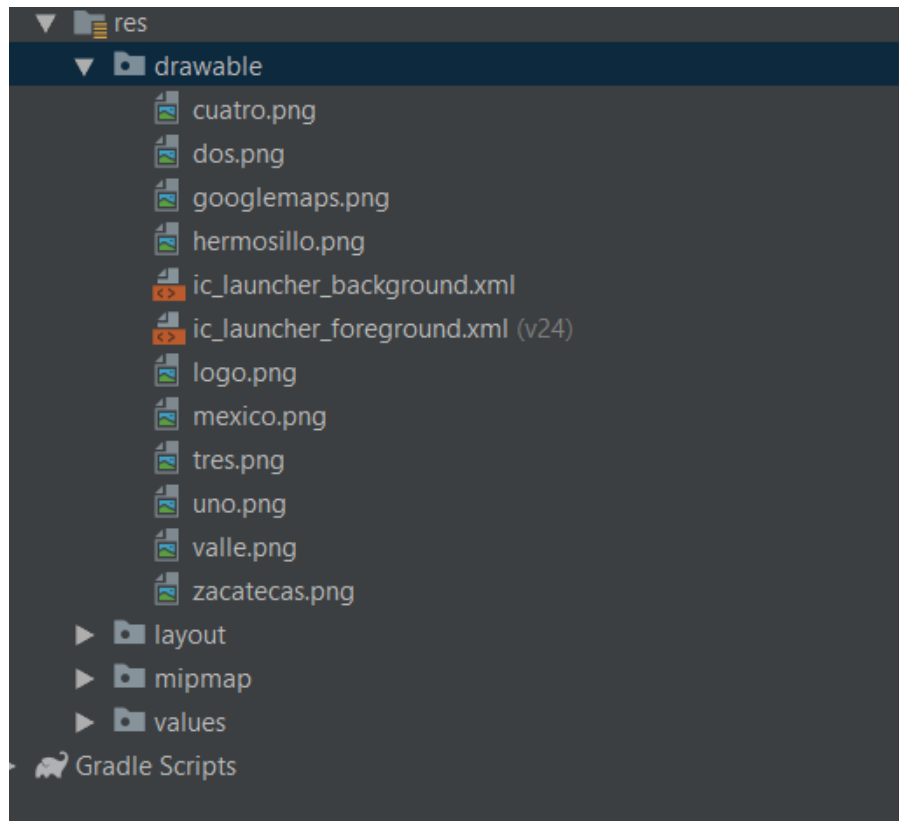


Diseñar interfaz

Para poder implementar nuestro mapa debemos tener un interfaz, para ello estaremos implementando la siguiente vista en Android studio.



Para poder implementar la imagen, debemos exportarla a nuestro *drawable*



Para mas facilidad se adjunta el código implementado para el diseño

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <ImageView
```



```
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:src="@drawable/googlemaps" />

    </LinearLayout>

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/fab_margin"
        android:text="@string/sitios" />

    <Button
        android:id="@+id/button3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/fab_margin"
        android:text="@string/type" />

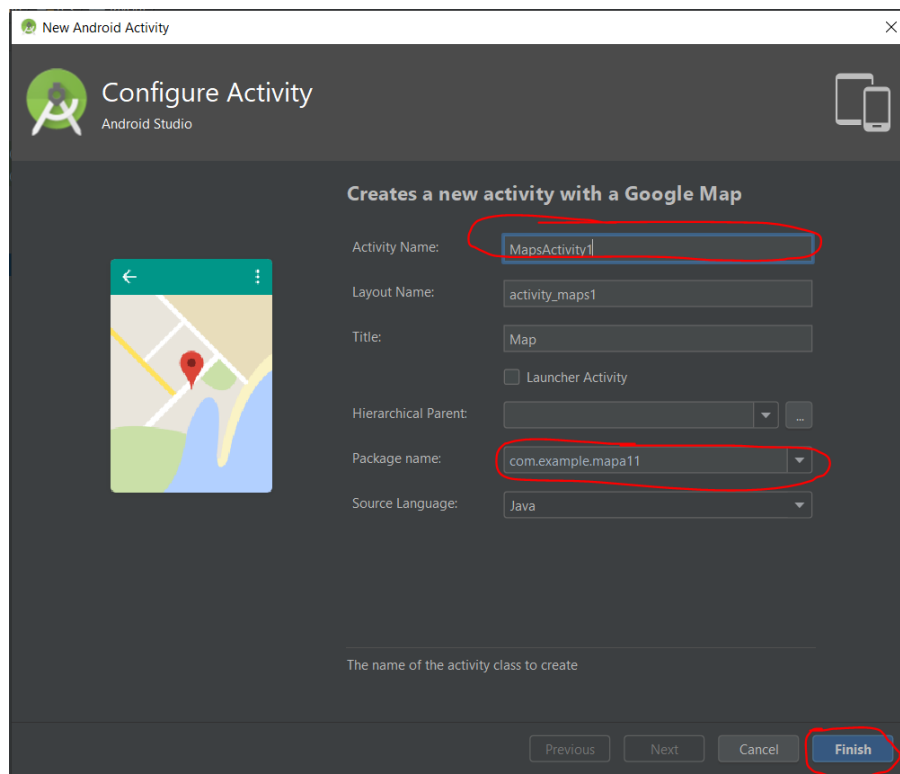
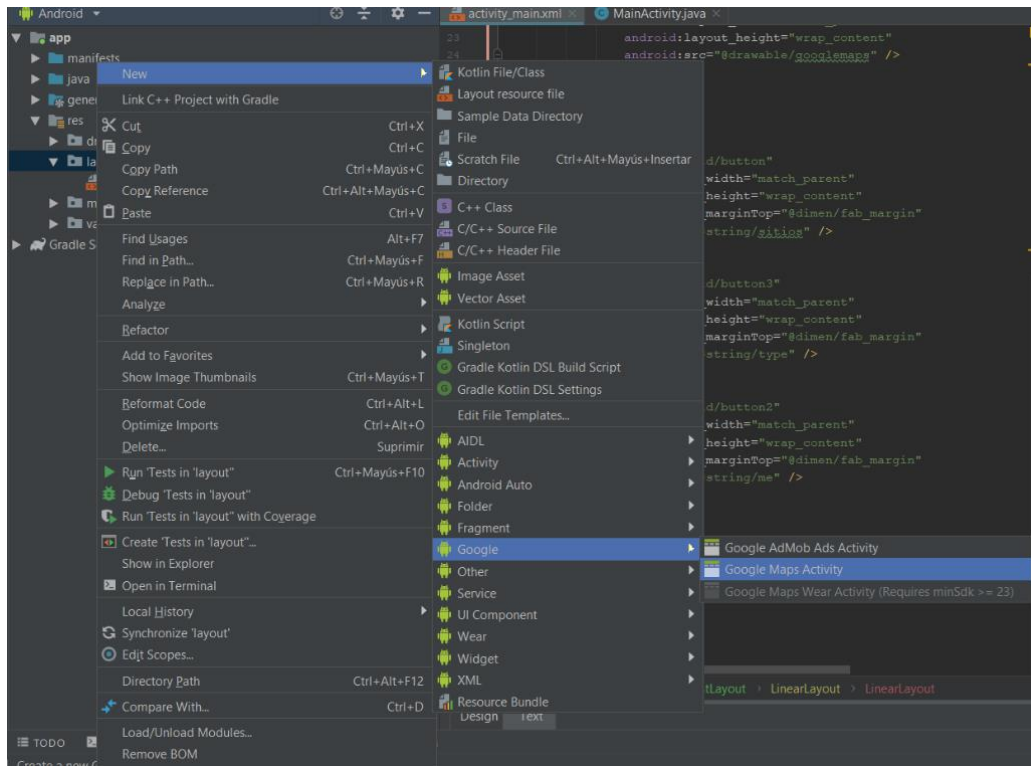
    <Button
        android:id="@+id/button2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/fab_margin"
        android:text="@string/me" />

    </LinearLayout>

</android.support.constraint.ConstraintLayout>
```

Crear API de Google

Debemos crear una nueva vista de la siguiente manera:



De esta forma estaremos implementando nuestra vista de Google Maps.

Para poder generar nuestro API vamos a necesitar nuestras credenciales que nos crea Android al crear nuestra vista de Google Maps.

```
TODO: Before you run your application, you need a Google Maps API key.

To get one, follow this link, follow the directions and press "Create" at the
https://console.developers.google.com/flows/enableapi?apiid=maps_android_back

You can also add your credentials to an existing key, using these values:

Package name:
C3:B7:1F:A2:4E:9C:78:69:F4:71:F8:38:F2:10:A7:C4:66:DC:71:91

SHA-1 certificate fingerprint:
C3:B7:1F:A2:4E:9C:78:69:F4:71:F8:38:F2:10:A7:C4:66:DC:71:91

Alternatively, follow the directions here:
https://developers.google.com/maps/documentation/android/start#get-key

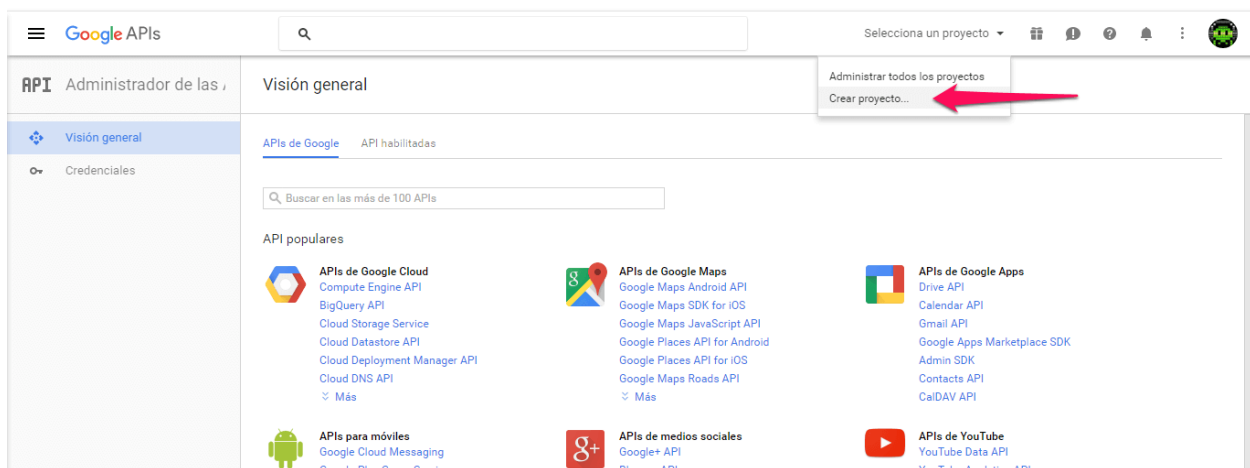
Once you have your key (it starts with "AIza"), replace the "google_maps_key"
string in this file.
-->
<string name="google_maps_key" templateMergeStrategy="preserve" translatable=
</resources>
```

Para usar una

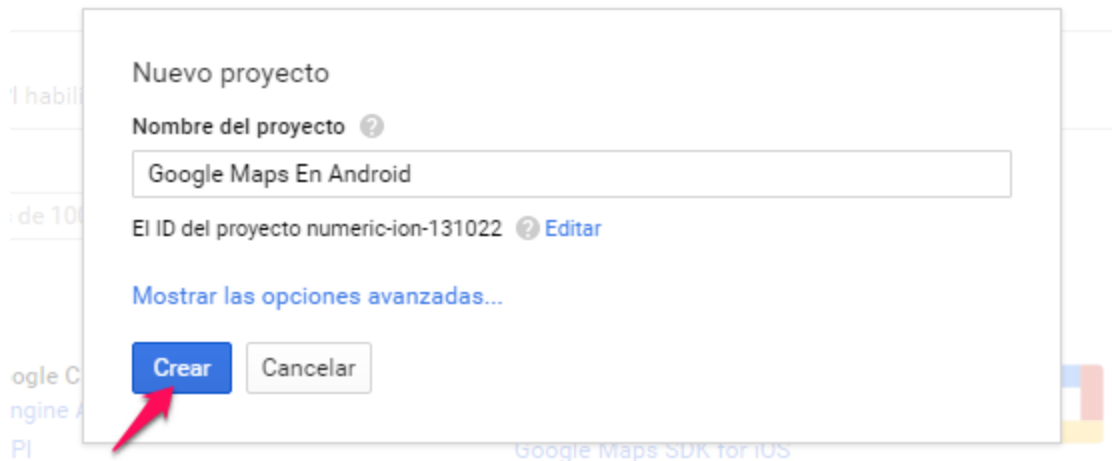
API de Google

es necesario crear un nuevo proyecto en Google Console Developers. Obviamente esto requiere que tengas una cuenta Google antes de manipular tu espacio.

1. Ingresa a tu Google Console Developers.
2. Selecciona el menú desplegable en la parte derecha de la toolbar que dice Selecciona un proyecto y selecciona Crear proyecto...

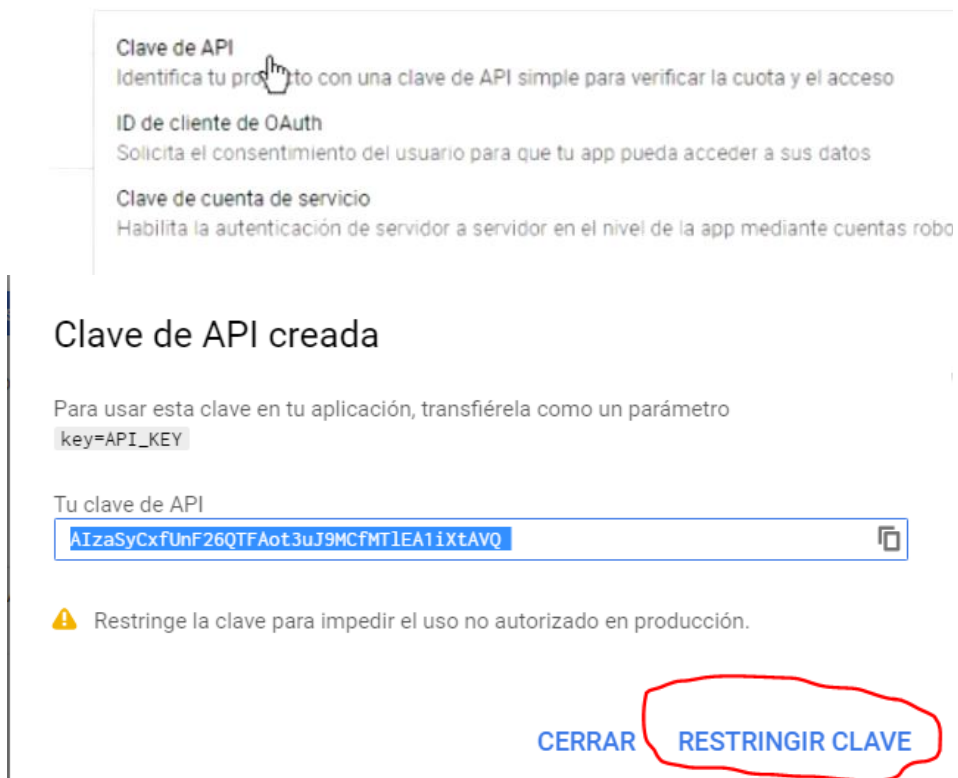


3. En el diálogo de creación asigna el nombre Google Maps En Android y confirma.



Si quieres puedes presionar Editar para cambiar el identificador de proyecto, por si tienes alguna convención de nombrado, ya que luego no será posible.

4. Luego ingresaremos las credenciales esta será nuestra clave de API



Restricciones de clave

Esta clave no tiene restricciones. Para evitar un uso sin autorización y el robo de cuotas, restringela. [Learn more](#)

⚠ Restricciones de aplicación: Ninguna ⚠ Restricciones de API: Ninguna

Restricciones de aplicación Restricciones de API

Las restricciones de aplicación especifican qué sitios web, direcciones IP o aplicaciones pueden usar esta clave. Puedes establecer un tipo de restricción por clave.

Restricciones de aplicación

- ☐ Ninguna
- ☐ URLs de referencia HTTP (sitios web)
- ☐ Direcciones IP (servidores web, tareas cron, etc.)
- ☒ **1** Aplicaciones para Android
- ☐ Aplicaciones para iOS

Restringir el uso a tus aplicaciones de Android (Opcional)

Añade el nombre del paquete y la huella digital del certificado de firma SHA-1 para restringir el uso de tus aplicaciones de Android.

Puedes encontrar el nombre del paquete en el archivo AndroidManifest.xml. A continuación, usa el comando siguiente para obtener la huella digital:

```
$ keytool -list -v -keystore mystore.keystore
```

+ Añadir nombre de paquete y huella digital

Ingresaremos nuestra credencial y nuestro paquete

Seguidamente guardamos. Habilitaremos las API de la siguiente manera

```
$ keytool -list -v -keystore mystore.keystore
```

Nombre de paquete

com.example.mapa11

Huella digital de certificado SHA-1

C3:B7:1F:A2:4E:9C:78:69:F4:71:F8:38:F2:10:A7:C4:66:DC:71:91

+ Añadir nombre de paquete y huella digital

API APIs y servicios **1**

Panel de control

Biblioteca

Credenciales


APIs y servicios

2 + HABILITAR APIS Y SERVICIOS



Todavía no tienes ninguna API que usar. Para empezar, haz clic en "Activar APIs y servicios" o ve a la [biblioteca de APIs](#).

Seguidamente habilitaremos los mapas para Android




Maps SDK para Android

Google

Mapas para su aplicación nativa de Android.

HABILITAR


De igual forma habilitaremos las siguientes



API de geocodificación

Google


Convertir entre direcciones y coordenadas geográficas.



API de geolocalización

Google

Datos de ubicación de torres celulares y nodos WiFi.



API de lugares

Google

Obtener información detallada sobre 100 millones de lugares.

Nos deberá quedar de la siguiente manera

APIs y servicios + HABILITAR APIS Y SERVICIOS

Panel de control

Biblioteca

Credenciales

No hay datos disponibles del periodo.

Ocultar API no utilizadas ?

Filtro

Nombre	↓ Solicitudes	Errores (%)	Latencia, mediana (ms)	Latencia del 95
API de geocodificación				
API de geolocalización				
Maps SDK para Android				
API de lugares				

Cargar el Mapa

Con la clave generada la sustituimos en el código

```
main.xml x | MapsActivity1.java x | google_maps_api.xml x | MainActivity.java x
resources
<!--
  TODO: Before you run your application, you need a Google Maps API key.

  To get one, follow this link, follow the directions and press "Create" at the end:
  https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID

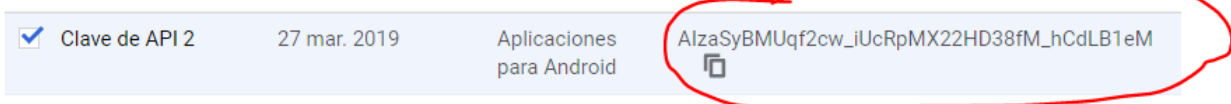
  You can also add your credentials to an existing key, using these values:

  Package name:
  C3:B7:1F:A2:4E:9C:78:69:F4:71:F8:38:F2:10:A7:C4:66:DC:71:91

  SHA-1 certificate fingerprint:
  C3:B7:1F:A2:4E:9C:78:69:F4:71:F8:38:F2:10:A7:C4:66:DC:71:91

  Alternatively, follow the directions here:
  https://developers.google.com/maps/documentation/android/start#get-key

  Once you have your key (it starts with "AIza"), replace the "google_maps_key"
  string in this file.
-->
<string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">
  YOUR_KEY_HERE</string>
</resources>
```



1. creamos nuestras variables de tipo button en MainActivity

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    final Button sitios = findViewById(R.id.sitios);
    final Button tipos = findViewById(R.id.tipos);
    final Button ubicacion =
    findViewById(R.id.ubicacion);
}
```

2. Inicializamos el botón

- a. Creamos una variable de tipo Intent, donde le tenemos que pasar 2
parametros, `getApplicationContext()` y el nombre de la clase, de la siguiente
manera.

```
sitios.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new
        Intent(getApplicationContext(), MapsActivity1.class):
    }
});
```

Para poder poner marcas en el mapa, debemos saber cual es la latitud y longitud del lugar, a continuación, se pondrá una lista con los lugares implementados con sus datos correspondientes:

Universidad Adventista de Centroamérica, Ruta Nacional 130, Provincia Alajuela, San Isidro, La Ceiba, 20106 Costa Rica

Latitud: 10.031557 | Longitud: -84.216883

Quiosco, Calle 2, Provincia Alajuela, Alajuela, El Carmen, 20101 Costa Rica

Latitud: 10.016429 | Longitud: -84.213793

Calle Loría, Provincia Alajuela, Tambor, 20112 Costa Rica

Latitud: 10.043365 | Longitud: -84.217205

Calle Tercera, Provincia Alajuela, San José, Pueblo Nuevo, 20102 Costa Rica

Latitud: 10.021166 | Longitud: -84.225722

Seguidamente después de tener nuestras coordenadas, abriremos nuestra MapsActivity1

Nuestro mapa trae consigo una marca predeterminada que está ubicada en Sydney.

```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    // Add a marker in Sydney and move the camera
    LatLng sydney = new LatLng(-34, 151);
    mMap.addMarker(new
MarkerOptions().position(sydney).title("Marker in Sydney"));

    mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
}
```

Este código se encuentra predeterminadamente en MapsActivity1, para nuestras marcas personalizadas solo usaremos las siguientes líneas de código, modificando a nuestro gusto el lugar con su respectiva latitud y longitud

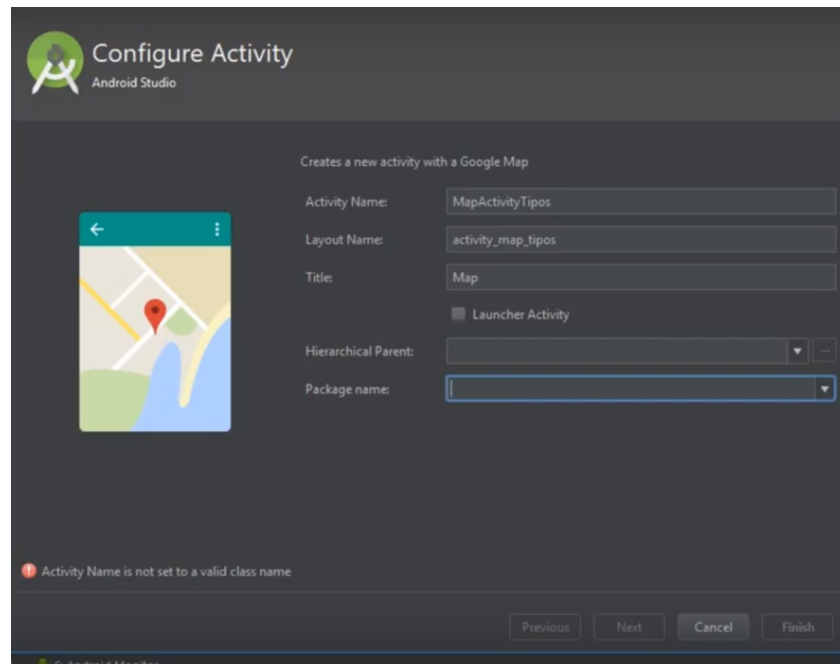
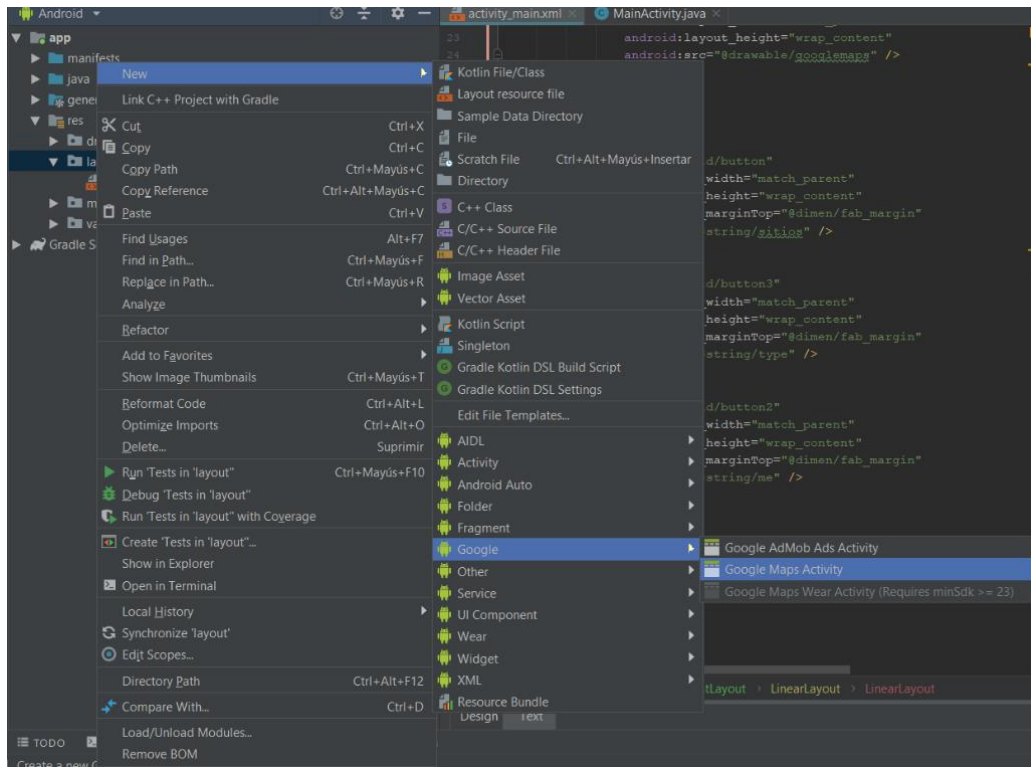
```
LatLng sydney = new LatLng(-34, 151);  
mMap.addMarker(new  
MarkerOptions().position(sydney).title("Marker in Sydney"));
```

Modificando los datos quedaría así

```
LatLng Unadeca = new LatLng(10.031557, -84.216883);  
mMap.addMarker(new  
MarkerOptions().position(Unadeca).title("UNADECA").snippet("Univ  
ersidad Adventista de Centroamérica"));  
mMap.moveCamera(CameraUpdateFactory.newLatLng(Unadeca));  
  
LatLng Quiosco = new LatLng(10.016429, -84.213793);  
mMap.addMarker(new  
MarkerOptions().position(Quiosco).title("Paque Central  
Alajuela").snippet("Quiosco, Calle 2, Provincia Alajuela,  
Alajuela, El Carmen, 20101 Costa Rica"));  
mMap.moveCamera(CameraUpdateFactory.newLatLng(Quiosco));  
  
LatLng CalleLoría = new LatLng(10.043365, -84.217205);  
mMap.addMarker(new  
MarkerOptions().position(CalleLoría).title("Casa  
James").snippet("Casa de James Ozuna"));  
mMap.moveCamera(CameraUpdateFactory.newLatLng(CalleLoría));  
  
LatLng PuebloNuevo = new LatLng(10.021166, -84.225722);  
mMap.addMarker(new  
MarkerOptions().position(PuebloNuevo).title("Casa  
James").snippet("Casa de James Ozuna"));  
mMap.moveCamera(CameraUpdateFactory.newLatLng(PuebloNuevo));
```

Tipos de mapas

Para poder visualizar diferentes tipos de mapas, debemos crear un nuevo layout



En el onCreate de nuestro MainActivity programamos el botón de tipos de mapas de nuestro diseño

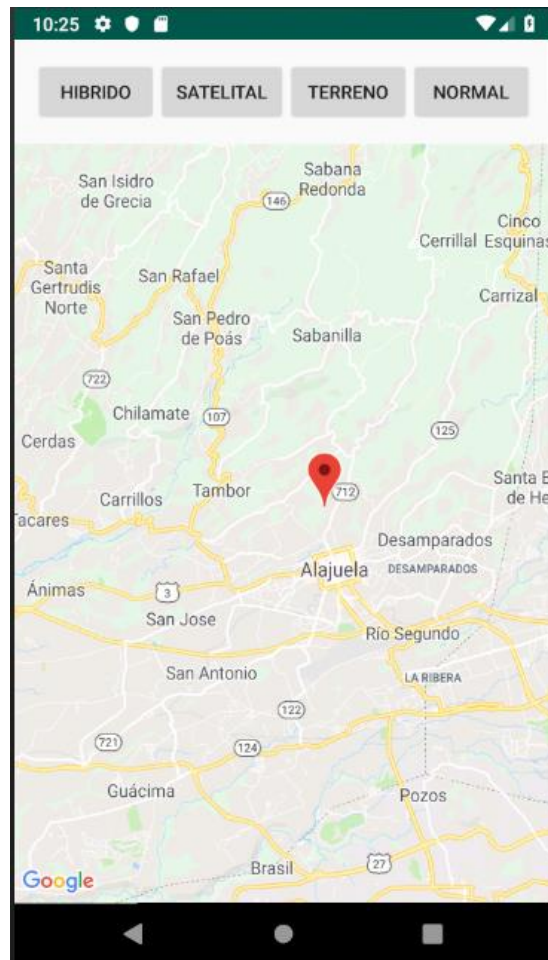
```
tipos.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        Intent intent = new  
Intent(getApplicationContext(), MapsActivityTipos.class);  
        startActivity(intent);  
    }  
});
```

De esta forma al oprimir nuestro botón, nos estará cargando el mapa, esto lo hacemos para ver si funciona bien.

Para poder trabajar la vista de los diferentes tipos de mapas, en nuestro MapsActivityTipos tenemos que crear un LinearLayout, el código queda de la siguiente manera.

```
<LinearLayout android:layout_height="match_parent"  
    android:layout_width="match_parent"  
    android:orientation="vertical"  
    xmlns:android="http://schemas.android.com/apk/res/android">  
  
    <fragment  
xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:map="http://schemas.android.com/apk/res-auto"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:id="@+id/map"  
  
    android:name="com.google.android.gms.maps.SupportMapFragment"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    tools:context="com.example.mapa11.MapsActivityTipos" />  
  
</LinearLayout>
```

Nuestro diseño de esta vista nos tendrá que quedar de la siguiente manera



```
<LinearLayout android:layout_height="match_parent"
    android:layout_width="match_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:padding="@dimen/fab_margin"
        >

        <Button
            android:id="@+id/Hibrido"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.25"
            android:text="@string/hibrido"
```

```

        android:textSize="11dp"/>

        <Button
            android:id="@+id/Satelital"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.25"
            android:text="@string/satelital"
            android:textSize="11dp"/>

        <Button
            android:id="@+id/terreno"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.25"
            android:text="@string/terreno"
            android:textSize="11dp"/>

        <Button
            android:id="@+id/normal"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.25"
            android:text="@string/normal"
            android:textSize="11dp"/>
    </LinearLayout>

    <fragment
xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:map="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:id="@+id/map"

        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context="com.example.mapa11.MapsActivityTipos" />
</LinearLayout>

```

Para implementar diferentes tipos de mapas nos vamos al MapsActivityTipos inicializamos nuestros botones ya creados anteriormente, y para cada uno creamos OnClickListener.

```
final Button hibrido = findViewById(R.id.Hibrido);
final Button satelital = findViewById(R.id.Satelital);
final Button terreno = findViewById(R.id.terreno);
final Button normal = findViewById(R.id.normal);

hibrido.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
    }
});

satelital.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);
    }
});

terreno.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN);
    }
});

normal.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
    }
});
```

Para programar el ultimo botón, creamos otra clase de la misma manera que hemos venido haciendo. En ActivityMain, para que nuestro botón cargar el mapa debemos crearlo

```
ubicacion.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new
Intent(getApplicationContext(), MapsActivitylocalizacion.class);
    }
});
```

Para añadir los botones de zoom solo basta con agregar una línea de código

```
mMap.getUiSettings().setZoomControlsEnabled(true);
```

Esto seria en MapsActivityLocalizacion

```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    LatLng Unadeca = new LatLng(10.031557, -84.216883);
    mMap.addMarker(new
MarkerOptions().position(Unadeca).title("UNADECA").snippet("Univ
ersidad Adventista de Centroamérica"));

mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(Unadeca, 12));

    mMap.getUiSettings().setZoomControlsEnabled(true);
}
```


Biografía

<https://cloud.google.com/maps-platform/>

<https://console.developers.google.com/apis/library?filter=category:maps&project=mapa-235807&folder&hl=es&organizationId&supportedpurview=project>

<https://www.coordenadas-gps.com/>